Optimal Testing/Maintenance Design in a Software Development Project

Koichiro Rinsaka
Department of Information Engineering
Hiroshima University
Higashi-Hiroshima
JAPAN
rinsaka@rel.hiroshima-u.ac.jp

Tadashi Dohi
Department of Information Engineering
Hiroshima University
Higashi-Hiroshima
JAPAN
dohi@rel.hiroshima-u.ac.jp

Abstract

This paper presents a stochastic model to determine both the optimal testing period and planned maintenance limit for computer software, considering the difference between the debugging environment in the testing phase and the executing environment in the operational phase. The software reliability models based on non-homogeneous Poisson processes are assumed to describe the debugging phenomena for both the environments. We formulate the total expected software cost, and derive the optimal software testing period and planned maintenance limit which minimize it. Throughout a numerical example, we calculate the joint optimal policy combined by testing period and planned maintenance limit.

1 Introduction

It is important to determine the optimal time when software testing should be stopped and when the system should be delivered to a user or a market. This problem, called *optimal software release problem*, plays a central role for the success or failure of a software development project. Many authors formulated the optimal software release problems based on different model assumptions and/or several software reliability models (Bai and Yun (1988), Dohi *et al.* (1997), Koch and Kubat (1983), Okumoto and Goel (1980), Yun and Bai (1990)).

It is common for software developers to provide maintenance service during the period when they are still responsible for fixing software faults causing failures. In order to carry out the maintenance in the operational phase, the software developer has to keep a software maintenance team. At the same time, the management cost in the operational phase should be reduced as much as possible, but the human resources should be utilized effectively.

Kimura *et al.* (1999) considered the optimal software release problem in the case where the software warranty period is a random variable. Pham and Zhang (1999) developed a software cost model with warranty and risk. They focused on the problem for determining when to stop the software testing under a warranty contract. Dohi *et al.* (2000) formulated the problem for determining the optimal software warranty period which minimizes the total expected software cost. Since the user's operational environment is not always same as that assumed in the software development phase, however, the above literature did not take account of the difference between two different phases.

Several reliability assessment methods during the operational phase have been proposed by some authors (Musa *et al.* (1996), Okamura *et al.* (2001)). In this paper, we develop a stochastic model on which the difference between the software testing environment and the operational environment is reflected in a fashion similar to Okamura *et al.* (2001). Based on the non-homogeneous Poisson processes (NHPPs), we formulate the total expected software cost incurred for the software developer, and derive analytically the optimal testing period (release time) which minimizes the total expected software cost. We call the time length to complete the operational maintenance after the release *a planned maintenance limit*, and also derive the optimal planned maintenance limit which minimizes the total expected software cost. Throughout a numerical example, we calculate numerically the joint optimal policy combined by testing period and planned maintenance limit.

2 Model description

First of all, the assumptions on the software fault-detection process are given in the following: (a) In each time when a software failure occurs, the software fault causing the failure is detected and removed immediately, (b) the number of initial faults contained in the software program, N_0, follows the Poisson distribution with mean ω (> 0), (c) the time to detect each software fault is independent and identically distributed nonnegative random variable with probability distribution function $F(t)$ and density function $f(t)$. Let $\{N(t), t \geq 0\}$ be the cumulative number of software faults detected up to time t. From the above assumptions, the probability math function of $N(t)$ is given by

$$\Pr\{N(t) = m\} = \frac{[\omega F(t)]^m e^{-\omega F(t)}}{m!} \quad m = 0, 1, 2, \cdots. \tag{1}$$

Hence, the stochastic process $\{N(t), t \geq 0\}$ is equivalent to an NHPP with mean value function $\omega F(t)$.

Suppose that a software testing is started at time 0 and terminated at time t_0 (≥ 0). The time length of software life cycle t_L (> 0) is known in advance and is assumed to be sufficiently larger than t_0. More precisely, the software life cycle is measured from time t_0. The software developer is responsible to the maintenance service for all the software failures that may occur during the software life cycle under a maintenance contract. We suppose that the project manager decides to break up the maintenance team at time $t_0 + t_W$ ($0 \leq t_W \leq t_L$) to reduce the cost to keep it, but he/she will incur a large amount of debugging cost after the planned maintenance limit compared with before it. Further, let c_0 (> 0) be the cost to remove each fault in the testing phase, c_W (> 0) be the cost to remove each fault before the planned maintenance limit, c_L (> 0) be the cost to remove each fault after the planned maintenance limit, k_0 (> 0) be the testing cost per unit of time, and k_W (> 0) be the cost to keep the maintenance team per unit of time.

3 Total expected software cost

This section formulates the total expected software cost which can occur in both testing and operational phases. It should be noted that the operational environment after the release may differ from the debugging environment in the testing phase. This difference is similar to that between the accelerated life testing environment and the normal operating environment for hardware products. We introduce the environment factor a (> 0) which expresses the relative severity in the operational environment after release, and consider that the time in the testing phase and the operational phase has proportional relationship. Namely, the time length t in the testing phase corresponds to at in the operational phase. Under the above assumptions, note that $a = 1$ means the equivalence between the testing and operational environments. On the other hand, $a > 1$ ($a < 1$) implies that the operational environment is severe (looser) than the testing environment. Okamura *et al.* (2001) apply this technique to model the operational phase of the software, and estimate the software reliability through an example of the actual software development project. The probability math function of the number of software faults detected before the planned maintenance limit is given by

$$\Pr\{N(t_0 + t_W) - N(t_0) = m\} = \frac{\{\omega [F(t_0 + at_W) - F(t_0)]\}^m}{m!} e^{-\omega[F(t_0+at_W)-F(t_0)]}. \tag{2}$$

Similarly, the fault-detection process of the software after the planned maintenance limit is expressed by

$$\begin{aligned} &\Pr\{N(t_0 + t_L) - N(t_0 + t_W) = m\} \\ &= \frac{\{\omega [F(t_0 + at_L) - F(t_0 + at_W)]\}^m}{m!} e^{-\omega[F(t_0+at_L)-F(t_0+at_W)]}. \end{aligned} \tag{3}$$

From Eqs.(2) and (3), the total expected software cost is given by

$$\begin{aligned} C(t_0, t_W) &= k_0 t_0 + c_0 \omega F(t_0) + k_W t_W + c_W \omega [F(t_0 + at_W) - F(t_0)] \\ &\quad + c_L \omega [F(t_0 + at_L) - F(t_0 + at_W)]. \end{aligned} \tag{4}$$

4 Determination of the optimal policies

Suppose that the time to detect each software fault obeys the exponential distribution (Goel and Okumoto 1979) with mean $1/\lambda$ (> 0). We make the following assumptions:

(A-I) $c_L > c_W > c_0$,
(A-II) $c_W \left(1 - e^{-\lambda a t_L}\right) > c_0$,
(A-III) $c_W \left(1 - e^{-\lambda a t_W}\right) + c_L \left(e^{-\lambda a t_W} - e^{-\lambda a t_L}\right) > c_0$.

Further, we define

$$Q(t_W) = k_0 + \omega\lambda \left[c_0 - c_W \left(1 - e^{-\lambda a t_W}\right) - c_L \left(e^{-\lambda a t_W} - e^{-\lambda a t_L}\right)\right]. \tag{5}$$

Then the following results provide the optimal software testing policy and optimal planned maintenance limit which minimize the total expected software cost, respectively.

Theorem 1: *When the software fault-detection time distribution follows the exponential distribution with mean $1/\lambda$, under the assumptions (A-I) to (A-III), the optimal software testing period (release time) which minimizes the total expected software cost is given as follows:*

(1) If $Q(t_W) < 0$, then there exists a finite and unique optimal software testing period t_0^* (> 0).

(2) If $Q(t_W) \geq 0$, then the optimal policy is $t_0^* = 0$ with

$$C(t_0^*, t_W) = k_W t_W + c_W \omega \left(1 - e^{-\lambda a t_W}\right) + c_L \omega \left(e^{-\lambda a t_W} - e^{-\lambda a t_L}\right). \tag{6}$$

Theorem 2: *When the software fault-detection time distribution follows the exponential distribution with mean $1/\lambda$, under the assumption (A-I), the optimal planned maintenance limit which minimizes the total expected software cost is given as follows:*

(1) If $k_W \geq (c_L - c_W)\omega\lambda a e^{-\lambda t_0}$, then the optimal policy is $t_W^* = 0$ with

$$C(t_0, t_W^*) = k_0 t_0 + c_0 \omega \left(1 - e^{-\lambda t_0}\right) + c_W \omega \left[e^{-\lambda t_0} - e^{-\lambda(t_0 + a t_L)}\right]. \tag{7}$$

(2) If $k_W < (c_L - c_W)\omega\lambda a e^{-\lambda t_0}$ and $k_W > (c_L - c_W)\omega\lambda a e^{-\lambda(t_0 + a t_L)}$, then there exists a finite and unique optimal planned maintenance limit t_W^* ($0 < t_W < t_L$) which minimizes the total expected software cost.

(3) If $k_W \leq (c_L - c_W)\omega\lambda a e^{-\lambda(t_0 + a t_L)}$, then we have $t_W^* = t_L$ with

$$C(t_0, t_W^*) = k_0 t_0 + c_0 \omega \left(1 - e^{-\lambda t_0}\right) + k_W t_L + c_W \omega \left[e^{-\lambda t_0} - e^{-\lambda(t_0 + a t_L)}\right]. \tag{8}$$

5 A numerical example

Based on 86 software fault data observed in the real software testing process (Abde-Ghaly *et al.* 1986), we calculate numerically the joint optimal policy (t_0^*, t_W^*). For the software fault-detection time distribution, we apply two distributions; exponential (Goel and Okumoto 1979) and gamma of order 2 (Yamada and Osaki 1985) distributions with parameter λ (> 0). Suppose that the unknown parameters in the software reliability models are estimated by the method of maximum likelihood, when 70 fault data are obtained, namely $t = 67.374$. Then, we have the estimates $(\hat{\omega}, \hat{\lambda}) = (98.5188,\ 1.84\text{e-}02)$ for the exponential distribution, and $(\hat{\omega}, \hat{\lambda}) = (75.1746,\ 6.46224\text{e-}02)$ for the gamma model. For the other model parameters, we assume: $k_0 = 0.02$, $k_W = 0.01$, $c_0 = 1.0$, $c_W = 2.0$, $c_L = 20.0$ and $t_L = 1000$.

Table 1 presents the dependence of the environment factor a on the joint optimal policy (t_0^*, t_W^*). It is observed from Table 1 that the optimal testing period t_0^* decreases as the environment factor monotonically increases, *i.e.*, the operational circumstance tends to be severe. It is also observed that its associated minimum total expected software cost $C(t_0^*, t_W^*)$ decreases as the environment factor monotonically increases.

Table 1: Joint optimal policy for varying environment factor.

	Exponential			Gamma		
a	t_0^*	t_W^*	$C(t_0^*, t_W^*)$	t_0^*	t_W^*	$C(t_0^*, t_W^*)$
0.50	405.0	0.0	107.7	167.4	0.0	78.9
0.75	304.6	159.2	107.3	135.6	50.4	78.7
1.00	282.6	157.1	106.8	128.5	49.8	78.6
1.25	272.7	143.3	106.5	125.2	45.5	78.5
1.50	267.0	129.8	106.2	123.4	41.2	78.4
2.00	260.6	108.4	105.9	121.3	34.3	78.3
3.00	254.9	81.5	105.5	119.4	25.8	78.2

References

Abde-Ghaly, A.A., Chan, P.Y. and Littlewood, B. (1986) Evaluation of competing software reliability predictions. *IEEE Trans. Software Eng. SE-12* 950–967.

Bai, D.S. and Yun, W.Y. (1988) Optimum number of errors corrected before releasing a software system. *IEEE Trans. Reliab. R-37* 41–44.

Dohi, T., Kaio, N. and Osaki, S. (1997) Optimal software release policies with debugging time lag. *Int. J. Reliab., Quality and Safety Eng. 4* 241–255.

Dohi, T., Okamura, H., Kaio, N. and Osaki, S. (2000) The age-dependent optimal warranty policy and its application to software maintenance contract. *Proc. 5th Int'l Conf. on Probab. Safe. Assess. and Mgmt.* (S. Kondo and K. Furuta, eds.), *4* 2547–2552, University Academy Press Inc.

Goel, A.L. and Okumoto, K. (1979) Time-dependent error-detection rate model for software reliability and other performance measures. *IEEE Trans. Reliab. R-28* 206–211.

Kimura, M., Toyota, T. and Yamada, S. (1999) Economic analysis of software release problems with warranty cost and reliability requirement. *Reliab. Eng. & Sys. Safe. 66* 49–55.

Koch, H.S. and Kubat, P. (1983) Optimal release time of computer software. *IEEE Trans. Software Eng. SE-9* 323–327.

Musa, J., Fuoco, G., Irving, N., Kropfl, D. and Juhlin, B. (1996) Chapter 5: The operational profile. in *Handbook of Software Reliability Engineering* (M.R. Lyu ed.), McGraw-Hill, New York.

Okamura, H., Dohi, T. and Osaki, S. (2001) A reliability assessment method for software products in operational phase — proposal of an accelerated life testing model —. *Electronics and Communication in Japan, Part 3 84* 25–33.

Okumoto, K. and Goel, L. (1980) Optimum release time for software systems based on reliability and cost criteria. *J. Sys. Software* **1** 315–318.

Pham, H. and Zhang, X. (1999) A software cost model with warranty and risk costs. *IEEE Trans. Computers 48* 71–75.

Yamada, S. and Osaki, S. (1985) Software reliability growth modeling: models and applications. *IEEE Trans. Software Eng. SE-11* 1431–1437.

Yun, W.Y. and Bai, D.S. (1990) Optimum software release policy with random life cycle. *IEEE Trans. Reliab., R-39* 167–170.